

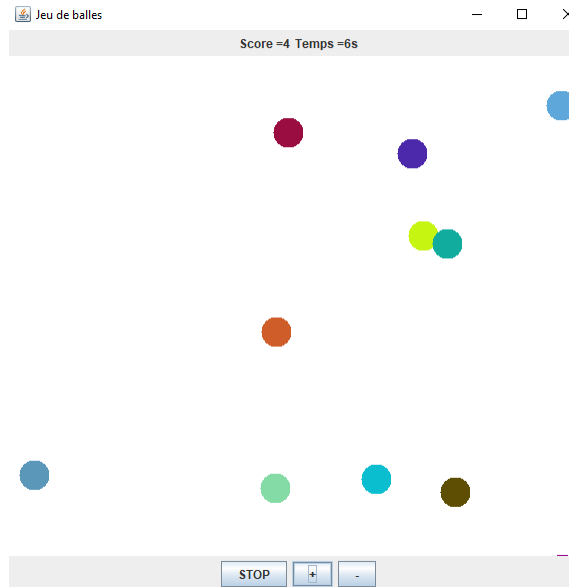
# Balles en mouvement

NICOLAS MAH-CHAK, L3 informatique

13 novembre 2018

## Résumé

Ceci est un rapport sur l'exercice des balles en mouvements. Dans cette exercice, nous devons créer une fenêtre qui affiche des balles en mouvements. Le principe de ce programme est simple, on compte le nombres de collisions entre chaque balles et à chaque collisions, les deux balles sont détruites.



## Introduction

Pour cette exercice, on doit créer une interface graphique pour animer des objets. En effet nous allons réaliser cela en Java. On va se servir de l'objet JFRAME mais aussi de la bibliothèque Swing de java pour nos animations comme les boutons. On utilisera aussi des threads pour afficher du texte.

# 1 Corps du rapport

## 1.1 Consignes

Ecrire un programme qui affiche des balles en mouvement qui respecte les consignes suivantes :

- Les balles rebondissent contre les bords et ne peuvent sortir de la fenêtre
- Un bouton START / STOP qui permet d'arreter le mouvement des balles
- Un bouton + qui nous permet d'ajouter une balle
- Un bouton - qui nous permet de retirer une balle
- Les balles qui rentrent entre collision sont détruites
- Chaque collision augmente le score de 1
- Un timer qui compte le temps lorsque les balles sont en mouvement et qui s'arrête lorsque les balles sont a l'arrêt.

## 1.2 Corps du programme

Mon programme se décompose en deux classes principales :

```
public class Fenetre extends JPanel {
    public Fenetre() {
        ...;
        ...;
        ...;
    }
}
```

Cette classe va gérer la création de la fenêtre, des boutons ect...

```
public class Rond {
    public Rond() {
        ...;
        ...;
        ...;
    }
}
```

Cette classe va gérer les ronds, leurs déplacements, leurs tailles, leurs positions ect...

## 1.3 Un exemple de fonction

```
public void go() {

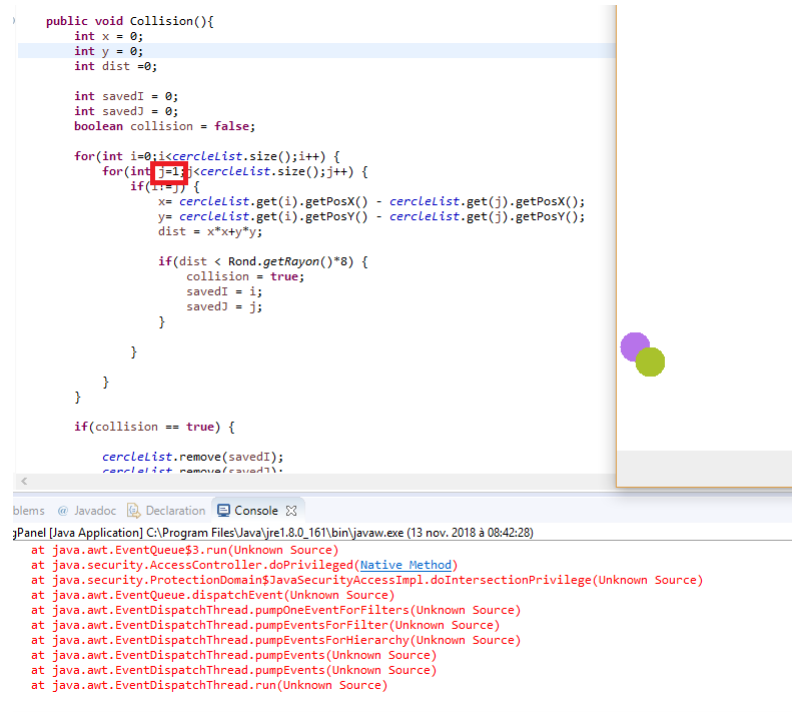
    if(Fenetre.getFlag()) {
        if (posX + vitessX < 0) {
            vitessX = Math.abs(vitessX);
        }
        if (posX + vitessX + RAYON >= dpWidth) {
            vitessX = -Math.abs(vitessX);
        }
        posX += vitessX;

        if (posY + vitessY < 0) {
            vitessY = Math.abs(vitessY);
        }
        if (posY + vitessY + RAYON >= dpHeight*0.83) {
            vitessY = -Math.abs(vitessY);
        }
        posY += vitessY;
    }
}
```

Cette fonction sert à déplacer une balle en fonction de sa position. Si elle sort de la fenêtre, elle change de direction pour faire un effet rebond. J'ai rajouté une vitesse aléatoire à chaque balle pour rendre plus dynamique le projet.

## 2 Problème et Conclusion

### 2.1 Quelques problèmes rencontrés



```
public void Collision(){
    int x = 0;
    int y = 0;
    int dist = 0;

    int savedI = 0;
    int savedJ = 0;
    boolean collision = false;

    for(int i=0;i<cercleList.size();i++) {
        for(int j=1;j<cercleList.size();j++) {
            if(i<j){
                x= cercleList.get(i).getPosX() - cercleList.get(j).getPosX();
                y= cercleList.get(i).getPosY() - cercleList.get(j).getPosY();
                dist = x*x+y*y;

                if(dist < Rond.getRayon()*8) {
                    collision = true;
                    savedI = i;
                    savedJ = j;
                }
            }
        }
    }

    if(collision == true) {
        cercleList.remove(savedI);
        cercleList.remove(savedJ);
    }
}
```

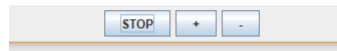
Panel [Java Application] C:\Program Files\Java\jre1.8.0\_161\bin\javaw.exe (13 nov. 2018 à 08:42:28)

```
at java.awt.EventQueue$3.run(Unknown Source)
at java.security.AccessController.doPrivileged(Native Method)
at java.security.ProtectionDomain$JavaSecurityAccessImpl.doIntersectionPrivilege(Unknown Source)
at java.awt.EventQueue.dispatchEvent(Unknown Source)
at java.awt.EventDispatchThread.pumpOneEventForFilters(Unknown Source)
at java.awt.EventDispatchThread.pumpEventsForFilter(Unknown Source)
at java.awt.EventDispatchThread.pumpEventsForHierarchy(Unknown Source)
at java.awt.EventDispatchThread.pumpEvents(Unknown Source)
at java.awt.EventDispatchThread.pumpEvents(Unknown Source)
at java.awt.EventDispatchThread.run(Unknown Source)
```

Pour faire ma fonction collision, je dois parcourir 2 fois ma liste avec 2 boucles for. Et si j'initialise j qui vaut 1 au lieu de 0, mon programme plante lorsqu'il reste uniquement 2 balles et que ses balles ont une collision.

```
cercleList.add(new Rond(new Color((int)(Math.random() * 0x1000000)), LONGUEUR, LARGEUR));
```

Pour mettre une couleur aléatoire à une balle, j'avais créer une liste de balles avec des couleurs différentes et puis je choisisais une balle dans cette liste au hasard et je la rajoutais cette balle dans la liste. Au final, j'ai eu quelque problème avec la manipulation des deux listes et j'ai donc décider d'ajouter une balles directement avec la methode color MATH.random pour avoir une couleur au hasard.



Une des contraintes du programme était le bouton start et stop. Quand on clique sur start, il devait passer en bouton stop. J'ai créer pour cela 2 bouton, un bouton start et un stop. Et chaque fois que j'utilisais le bouton start, il disparaissait et on voyait le bouton stop. Mais je n'arrivais pas à le faire disparaître complètement. C'est pour sa que j'ai utilisé qu'un seul bouton que je modifie chaque fois que je l'utilise.

## 2.2 Conclusion

La réalisation de ce programme fait appel à la notion de threads, à la réalisation de fenêtre, de boutons ect... Cette exercice ma permis de revoir comment ajouter des boutons, leurs fonctionnalités et surtout comment bien les placer, de manipuler les objets graphiques avec leur positions. Si j'avais plus de temps j'aurais aimé approfondir l'aspect collision des balles. Si au lieu de disparaître elle rebondissait entre elle. J'aurai pu ajouter aussi des figures autres que des ronds. J'ai choisi de faire ce projet en java parce que c'est le langage qui me parait le plus accessible pour moi.

## 3 Référence

1. <https://openclassrooms.com/fr/courses/26832-apprenez-a-programmer-en-java/23193-le-fil-rouge-une-animation>
2. <http://zetcode.com/tutorials/javagamestutorial/collision/>
3. <https://openclassrooms.com/fr/courses/26832-apprenez-a-programmer-en-java/23366-positionner-des-boutons>
4. <https://gamedev.stackexchange.com/questions/62921/java-collision-detection-with-rectangles>
5. <https://stackoverflow.com/questions/345838/ball-to-ball-collision-detection-and-handling>